Using Tree Kernels for Classifying Temporal Relations between Events

Seyed Abolghasem Mirroshandel, Gholamreza Ghassem-Sani, and Mahdy Khayyamian

Department of Computer Engineering, Sharif University of Technology
Azadi Ave., 11155-9517 Tehran, Iran
mirroshandel@ce.sharif.edu, sani@sharif.edu, khayyamian@ce.sharif.edu

Abstract. The ability to accurately classify temporal relations between events is an important task in a large number of natural language processing and text mining applications such as question answering, summarization, and language specific information retrieval. In this paper, we propose an improved way of classifying temporal relations, using support vector machines (SVM). Along with gold-standard corpus features, the proposed method aims at exploiting useful syntactic features, which are automatically generated, to improve accuracy of the SVM classification method. Accordingly, a number of novel kernel functions are introduced and evaluated for temporal relation classification. Our evaluations clearly demonstrate that adding syntactic features results in a considerable performance improvement over the state of the art method, which merely employs gold-standard features.

Keywords: Temporal Relations between Events, Classification, Information Retrieval, Text Mining

1 Introduction

In recent years, many progresses have been made in natural language processing (NLP). Combining statistical and symbolic methods plays a significant role in these advances. Tasks such as part-of-speech tagging, morphological analysis, parsing, and named entity recognition have been addressed with satisfactory results (Mani et al., 2006). Problems such as temporal information processing that requires a deep semantic analysis are yet to be addressed.

Lately, the increasing attention in practical NLP applications such as question answering, information extraction, and summarization have resulted in an increasing demand for temporal information processing (Tatu and Srikanth, 2008). In question answering, one would expect the system to answer questions such as "when an event occurred", or "what is the chronological order between some desired events". In text summarization, especially in multi-document type, knowing the order of events is a useful source for merging related information correctly. It is also the case that in some information extraction applications, the temporal information between events can be very useful and effective (Alonso, 2009).

Temporal information is usually encoded in the textual description of some events. For a given ordered pair of components (x_1, x_2), where x_1 and x_2 are times or events, a temporal information processing system tries to identify the type of relation r_i that temporally links x_1 to x_2. The type of relation r_i can be one of the 13 types proposed by James Allen (Allen, 1984). For example, in the sentence *"Ocean Drilling said (e22) it will offer (e23) 15% to 20% of the contract-drilling business through an initial public offering (e25) in the near future (t67).* (wsj_313), there are some relations between pairs (e23, e25), and (e25, t67). The task is to automatically tag these pairs with relations INCLUDES and BEFORE, respectively.

With recent construction of the Timebank corpus (Pustejovsky et al, 2003), the efficiency of different machine learning methods can now be compared. The recent work with Timebank has disclosed that six-class classification of temporal relations is a very complicated task, even for human annotators. In this paper, we propose an improved way of classifying temporal relations, using a machine learning approach. Support vector classification using effective kernel functions are specifically applied to two types of features: corpus gold-standard event features and underlying syntactic features of the contextual sentence. To capture either type of features, we apply an event-kernel to the gold-standard event features, and a convolution tree-kernel to syntactic features. The event kernel has been implemented according to (Mani et al., 2006) and some novel tree kernels have been employed as our syntactic tree kernel. Experimental results on Timebank validate the proposed method by showing 6% improvement over the state of the art method that merely uses gold-standard features.

The remainder of the paper is organized as follows: section 2 is about previous approaches to temporal relation classification. Section 3 explains our proposed method. Section 4 briefly presents characteristic of the corpus that we have used. Section 5 demonstrates evaluation of the proposed algorithm. Finally, paper is concluded in section 6.

2 Previous Works

There are numerous ongoing researches focused on temporal relation classification. These efforts can be divided into three categories: 1) Pattern based; 2) Rule based, and 3) Anchor based. These categories are discussed in the next three sub-sections.

2.1 Pattern Based Methods

This group of methods tries to extract some generic lexico-syntactic patterns for events co-occurrence. Extracting these patterns can be done manually or automatically.

Manual Extraction of Patterns

Perhaps the simplest pattern based method is the one that was developed using a knowledge resource called VerbOcean (Chklovski and Pantel, 2005). VerbOcean has a small number of manually selected generic patterns. The style of patterns is in the form of <Verb-X> and then <Verb-Y>. After manually creating these patterns, this method can obtain some of existing semantic relations between events. Similar to other manual methods, a major drawback of this method is its tendency to have a high recall but a low precision. One way to overcome this weakness is to create more specific patterns; however it is clear that this would be very hard and time consuming.

Another way of resolving the low precision problem is using an additional component for pruning extracted relations. Many researches have tried to address this issue by a variety of approaches. In some studies, several heuristics have been employed to resolve the low precision problem (Chklovski and Pantel, 2005; Torisawa, 2006). Another solution is incorporating a classifier that is trained on a related corpus (Inui et al., 2003) and is used to refine the results.

Automatic Extraction of Patterns

These methods use machine learning techniques for pattern extraction. They try to learn a classifier from an annotated corpus, and attempt to improve classification accuracy by feature engineering.

MaxEnt classifier is a good example of this group (Mani et al., 2006). MaxEnt assigns one of six relations to each pair of events from an augmented Timebank corpus. This classifier uses perfect features, which have been hand-tagged in the corpus, including tense, aspect, modality, polarity, and event class. In addition to these features, it relies on two additional features including pairwise agreement of tense and aspect. In this paper, we propose a new technique to improve this particular method.

There is another approach in this group that trains an event classifier for intra-sentential events, and builds a corpus that contains sentences with at least two events, one of which is triggered by a key time word (e.g., after, before, etc.). The classifier is based on syntax and clausal ordering features (Lapata and Lascarides, 2006).

The state of the art in this group is very similar to the MaxEnt classifier. It relies on features extracted automatically from some raw text, and works 3% better than MaxEnt. This classifier tries to learn event attributes and event-event features in two consecutive stages. Event attributes are the same as that of MaxEnt, but event-event features are new and include part of speeches, event-event syntactic properties, prepositional phrase, and temporal discourses (Chambers et al, 2007). This method also uses some extra resources like WordNet to find words' synsets.

There are also other methods that have used some machine learning techniques for acquisition of semantic relations between events (Abe et al., 2008). Such techniques can be applied to temporal relation classification as well. In addition to these methods, there is an SVM-based method which has been shown satisfactory results in event-time relation classification (Mirroshandel et al., 2009).

2.2 Rule Based Methods

The common idea behind rule based methods is to find some rules for classifying temporal relations. In most existing works, these rules are determined manually and are based on Allen's interval algebra (Allen, 1984).

In a study, rules of temporal transitivity were applied to increase the training set by a factor of 10. Next, the MaxEnt classifier was trained on this enlarged corpus. The test accuracy on this enlarged corpus was very encouraging. There was nearly 32% progress in accuracy (Mani et al., 2006).

Reasoning with determined rules is another usage of rules. In (Tatu and Srikanth, 2008), a rich set of rules (axioms) was created. Then by using a first order logic based theorem prover, they tried to find a proof of each temporal relation by refutation.

2.3 Anchor Based Methods

Anchor based methods use information of argument fillers (i.e., anchors) of every event expression as a valuable clue for recognizing temporal relations between events. They are based on the distributional hypothesis (Harris, 1968) and by looking at a set of event expressions whose argument fillers have a similar distribution, they try to recognize synonymous event expressions. Algorithms such as DIRT (Lin and Pantel, 2001), TE/ASE (Szpektor et al., 2004), and that of Pekar's system (Pekar, 2006) are some examples of anchor based methods.

It has been shown that one can gain more accuracy by combining some of these three different methods. For example, pattern and rule based methods were merged (Mani et al., 2006), and the new system showed to be more efficient than each of the base methods. In the other study, pattern and anchor based methods were combined (Chklovski and Pantel, 2005; Abe et al., 2008). However, there has been an exception: merging pattern and anchor based methods did not gain any improvement (Torisawa, 2006).

3 Tree Kernel Based Temporal Relation Classification

Syntactic features have been shown to be a great source of information in various NLP and text mining applications such as relation extraction, semantic role labeling, and co-reference resolution. Current works in temporal relation classification have not sufficiently utilized such features. Here, we aim at taking advantage of syntactic features. Because of promising results of Support Vector Machines (SVM) (Boser et al., 1992; Cortes and Vapnik 1995) in related works, it has been chosen as our classification algorithm. To incorporate syntactic features into SVM, convolution tree kernels are applied. More specifically, these tree kernels have been

combined with a simple event kernel. In the next sub-section, the simple event kernel is briefly discussed. Then the convolution tree kernels are described, followed by the explanation of ways of combining these kernels.

3.1 Simple Event Kernel

This is a linear kernel that exclusively uses gold-standard features of events. For each event, there are five temporal attributes which have been tagged in Timebank: 1) tense; 2) grammatical aspect; 3) modality; 4) polarity, and 5) event class. Tense and grammatical aspect define temporal location and event structure; thus, they are necessary in any method of temporal relation classification. Modality and polarity specify non-occurring (or hypothetical) situations. The event class shows the type of event. The range of values for these attributes is based on (Pustejovsky et al, 2003).

In addition to these five attributes, it uses part of speech tags of event as an extra feature. This kernel can be defined as follows:

$$K_{TR}(TR_1, TR_2) = \sum_{i=1,2} K_E(TR_1.E_i, TR_2.E_i) \quad (1)$$

where TR_1 and TR_2 stand for two temporal relation instances, E_i is the i^{th} event of a temporal relation instance, and K_E is a simple kernel function over the features of event instances:

$$K_E(E_1, E_2) = \sum_i C(E_1.f_i, E_2.f_i) \quad (2)$$

where f_i means the i^{th} event feature; function *C* returns 1 if the two feature values are identical, and returns 0 otherwise. In essence, K_E returns the number of common feature values of two event instances.

3.2 Tree kernels

In (Khayyamian et al., 2009), a generalized version of convolution tree kernel (Collins and Duffy, 2001) was proposed by associating generic weights to the nodes and sub-trees of the parse tree. In this paper, some customized versions of this kernel are used to capture syntactic features.

Generalized Convolution Tree Kernel

A generalized convolution tree kernel was proposed in (Khayyamian et al., 2009). In order to explain the kernel, first a feature vector over the parse tree is defined in equation (3). In this vector, the i^{th} feature equals to the sum of weighted number of occurrences of sub-tree type i^{th} in the parse tree.

Function $I_{subtree_i}(n)$ is an indicator that returns 1 if $subtree_i$ occurs at node n, and returns 0 otherwise. $subtree_i(n)$ is the sub-tree instance of type i^{th} which is rooted in node n. As it is shown in equation (4), function *tw(T)* (which denotes "tree weight") assigns a weight to tree T, which is the product of all its node weights. *in(T)* and *en(T)* are respectively sets of internal and external nodes of *T*.

$$\begin{aligned} H(T) = (&\sum_{n\in T}[I_{subtree_1}(n) \times tw(subtree_1(n))],..., \\ &\sum_{n\in T}[I_{subtree_i}(n) \times tw(subtree_i(n))],..., \\ &\sum_{n\in T}[I_{subtree_m}(n) \times tw(subtree_m(n))]) \end{aligned} \quad (3)$$

$$tw(T) = \prod_{n \in in(T)} inw(n) \times \prod_{n \in en(T)} enw(n) \tag{4}$$

$$K(T_1, T_2) = \langle H(T_1), H(T_2) \rangle \tag{5}$$

$$= \left(\sum_i \sum_{n_1 \in T_1} I_{subtree_i}(n_1) \times tw(subtree_i(n_1)) \right) \times$$

$$\left(\sum_i \sum_{n_2 \in T_2} I_{subtree_i}(n_2) \times tw(subtree_i(n_2)) \right)$$

$$= \sum_{n_1 \in T_1} \sum_{n_2 \in T_2} [\sum_i I_{subtree_i}(n_1) \times I_{subtree_i}(n_2) \times$$

$$tw(subTree_i(n_1)) \times tw(subTree_i(n_2))]$$

$$= \sum_{n_1 \in T_1} \sum_{n_2 \in T_2} C(n_1, n_2)$$

Because each node of the entire parse tree can either occur as an internal or as an external node of a specific sub-tree (provided that it exists in the sub-tree), two weighted types are respectively associated with the nodes by *inw(n)* and *enw(n)* functions (these stand for "internal node weight" and "external node weight"). For example, in Figure 1, while the node with label PP is an external node of sub-trees (1) and (7), it is regarded as an internal node of sub-trees (3) and (4).

As shown in equation (5), a method similar to that of (Collins and Duffy, 2001) can be employed to devise a kernel function for the calculation of dot products of *H(T)* vectors. According to equation (5), the calculation of the kernel eventually leads to the sum of function $C(n_1, n_2)$ over all tree node pairs T_1 and T_2. Function $C(n_1, n_2)$ is the weighted sum of common sub-trees n_1 and n_2, and can be recursively calculated (similar to function $C(n_1, n_2)$ in (Collins and Duffy, 2001)).

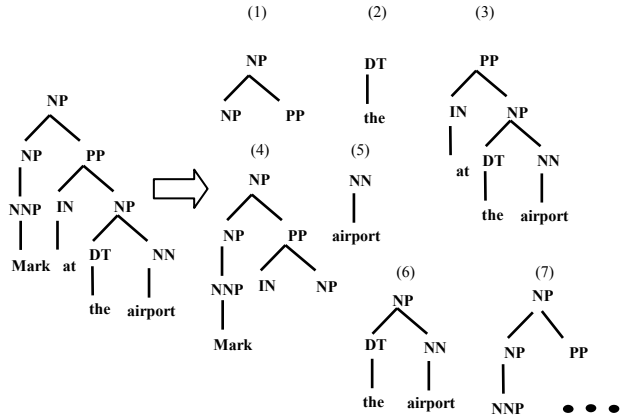

Figure 1: Samples of sub-trees used in convolution tree kernel calculation.

Kernel Customization for Temporal Relation Classification

In (Khayyamian et al., 2009), four sub-kernels of the generalized convolution tree kernel were proposed. It seems that these kernels can be applied to temporal relation classification. Using weighting functions of the generalized kernel, the customized kernels differentiate among sub-trees based on how their nodes interact with the event arguments.

Since the whole syntactic parse tree of the sentence that holds the event arguments contains plenty of misleading features, as in (Zhang et al., 2006), Path-enclosed Tree (PT) is chosen as

our tree portion for applying tree kernels. PT is a portion of parse tree that is enclosed by the shortest path between two event arguments.

- **The Original Collins and Duffy Kernel**

By setting $inw(n) = \alpha = \sqrt{\lambda}$ and *enw(n)=1* for all nodes, the generalized kernel can be converted to the kernel proposed in (Collins and Duffy, 2001). In their paper, parameter $0 < \lambda \leq 1$ is a decaying parameter used to retain the kernel values within a fairly small range. Without this parameter, the value of the kernel for identical trees becomes much larger than its value for different trees, which slows down SVM convergence.

- **Argument Ancestor Path Kernel (AAP)**

Definition of weighting functions is as follows. Parameter $0 < \alpha \leq 1$ is a decaying parameter analogous to λ. This weighting method is equivalent to applying original Collins and Duffy kernel on a portion of the parse tree that exclusively includes the arguments ancestor nodes and their direct children.

$$inw(n) = \begin{cases} \alpha & \text{if } n \text{ is on the argument ancestor path} \\ & \text{or a direct child of a node on it} \\ 0 & \text{otherwise} \end{cases} \quad (6)$$

$$enw(n) = \begin{cases} 1 & \text{if } n \text{ is on the argument ancestor path} \\ & \text{or a direct child of a node on it} \\ 0 & \text{otherwise} \end{cases} \quad (7)$$

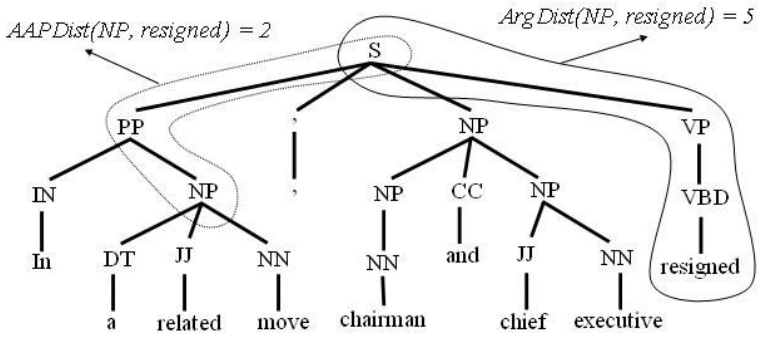

Figure 2: A syntactic parse tree with AAPDist and ArgDist example. There is a SIMULTAENOUS temporal relation between (move, resign) event pair in this parse tree.

- **Argument Ancestor Path Distance Kernel (AAPD)**

Weighting functions are defined in the following equations. Both functions have identical definitions for this kernel.

$$inw(n) = enw(n) = \alpha^{\frac{\min(AAPDist(n,\ \arg_1),\ AAPDist(n,\ \arg_2))}{MAXDIST}} \quad (8)$$

Function *AAPDist(n, arg)* computes the distance of node n from ancestor path of event argument *arg* on the parse tree as depicted in Figure 2. *MAXDIST* is used for normalization, and is the maximum value of *AAPDist* in the whole tree. Using this weighting approach, the closer a node is to one of the arguments ancestor path, the less it is decayed by the weighting function.

- **Argument Distance Kernel (AD)**

Weighting functions of this kernel, which have identical definitions, are shown as follows. Their definitions are similar to the previous kernel functions, though they use a different distance function which measures the distance of a node from an event argument rather than its ancestor path (see Figure 2).

$$inw(n) = enw(n) = \alpha^{\frac{\min(ArgDist(n,\ \arg_1),\ ArgDist(n,\ \arg_2))}{MAXDIST}} \tag{9}$$

- **Threshold Sensitive Argument Ancestor Path Distance Kernel (TSAAPD)**

This kernel is intuitively similar to AAPD kernel; except that instead of using a smooth decaying method, it employs a threshold based technique. Weighting functions are as follows:

$$inw(n) = enw(n) = \begin{cases} 1 & AAPDist(n) \le Threshold \\ \alpha & AAPDist(n) > Threshold \end{cases} \tag{10}$$

3.3 Composite Kernels for Temporal Relation Extraction

In this section, two types of composition are proposed: linear composition and polynomial composition (Zhang et al., 2006).

Linear Composite Kernel

$$K_l(TR_1, TR_2) = \alpha \hat{K}_1(TR_1, TR_2) + (1 - \alpha)\hat{K}_2(TR_1, TR_2) \tag{11}$$

where $\hat{K}_1$ can be a normalized form of one of the mentioned convolution tree kernels. A kernel *K(X, Y)* can be normalized by dividing it by $\sqrt{K(X,X).K(Y,Y)}$. $\hat{K}_2$ is the normalized form of simple event kernel. α is the composition coefficient. Based on five tree kernels that have been introduced, five linear composite kernels can be accordingly produced.

Polynomial Composite Kernel

$$K_P(TR_1, TR_2) = \alpha \hat{K}_1(TR_1, TR_2) + (1 - \alpha)\hat{K}_2{}^P(TR_1, TR_2) \tag{12}$$

where $\hat{K}_1$, $\hat{K}_2$ and α have similar definition as in linear composite kernel. $\hat{K}_2{}^P$ is the polynomial expansion of $\hat{K}_2$ with degree d (in this work, we have assumed *d=2*) and is defined as follows:

$$\hat{K}_2{}^P = (1 + \hat{K}_2)^d \tag{13}$$

Five different polynomial composite kernels can also be constructed in this case.

4 Corpus Description

We used Timebank (v 1.2) with 183 newswire documents and 64077 words, and for comparison with previous works, we added 73 documents of the Opinion Corpus (Mani et al., 2006), which has 38709 words. These two datasets have been released based on TimeML (Pustejovsky et al, 2003). There are 14 temporal relations in TLink (Event-Event and Event-

Time relations) class of TimeML. Similar to (Mani et al., 2006; Tatu and Srikanth, 2008; Mani et al., 2007), we used a normalized version of these 14 temporal relations that contains 6 temporal relations RelTypes = {SIMULTANEOUS, IBEFORE, BEFORE, BEGINS, ENDS, INCLUDES}. For converting 14 relations to 6, the inverse relations were omitted, and SIMULTAENOUS and IDENTITY, as well as DURING and IS_INCLUDED, were collapsed.

In our experiments, we merged two Timebank and Opinion datasets to generate a single corpus called OTC. Table 1 shows the normalized TLink class distribution over OTC.

Table 1: The normalized Event-Event TLink distribution in the Timebank and OTC.

Relation	Timebank	OTC
IBEFORE	63 (1.8 %)	131 (2.13 %)
BEGINGS	77 (2.21 %)	160 (2.60 %)
ENDS	114 (3.27 %)	208 (3.38 %)
SIMULTANEOUS	1304 (37.46 %)	1528 (24.86 %)
INCLUDES	588 (16.89 %)	950 (15.45 %)
BEFORE	1335 (**38.35 %**)	3170 (**51.57 %**)
TOTAL	**3481 (2387 intra-sentential)**	**6147 (4377 intra-sentential)**

As it is shown in table 1, relation "BEFORE" is the most frequent relation; thus it forms the majority class, and has been used as the baseline of experiments.

5 Experiments

We have used LIBSVM (Chang and Lin, 2001) java source for the SVM classification (one-versus-one multi class strategy), Stanford NLP package (available at http://nlp.stanford.edu/software/index.shtml) for tokenization, sentence segmentation, and parsing.

Since tree kernels can be more appropriately applied to the event pairs that reside on the same sentence, the corpus data have been accordingly split into two intra-sentential and inter-sentential parts. The proposed kernels have been evaluated on the intra-sentential instances, while the simple event kernel has been exclusively used for the inter-sentential instances. The results reported for the whole corpus has been produced by combining those results. All the results are the outcome of a 5-fold cross validation. In order to find the appropriate value for parameters, 1000 event pairs have been randomly chosen as development set.

Table 2 shows the accuracy results of employing different tree kernels. In our evaluation, baseline was the majority class (BEFORE relation) of the evaluated corpus. Mani is the state of the art method, which exclusively uses gold-standard features (Mani et al., 2006). The other methods were described in the subsection 3.2.

The results show that using syntactic structure of sentences can be very effective. Comparing with other methods, AAPD kernel has achieved the best results. It showed 3% improvement over Mani's method on Timebank and 1% over OTC. The other tree kernels showed satisfactory results, too.

Table 2: The accuracy of tree kernels on Timebank and OTC.

Method	Timebank Corpus	OTC Corpus
Baseline	38.35	51.57
Mani	50.97	62.5
CollinsDuffy	51.71	62.04
AAP	53.41	62.52
AAPD	**54**	**63.44**
AD	53.3	62.38
TSAAPD	53	62.53

As it is demonstrated in table 3, the effective exploitation of syntactic and simple event features in the linear composite kernels (subsection 3.3) resulted in a noticeable improvement of accuracy. Here, AAPD linear composite kernel was the most successful kernel, which gained over 6% improvement on Timebank, and 3% progress in accuracy on OTC.

Table 3: The accuracy of linear composite kernels on Timebank and OTC.

Method	Timebank Corpus	OTC Corpus
CollinsDuffy Linear	56.67	65.27
AAP Linear	56.12	64.88
AAPD Linear	**56.73**	**65.62**
AD Linear	56.6	65.34
TSAAPD Linear	56.4	65.24

Table 4 shows the accuracy results of applying five polynomial composite kernels (subsection 3.3) to Timebank and OTC.

Table 4: The accuracy of polynomial composite kernels on Timebank and OTC.

Method	Timebank Corpus	OTC Corpus
CollinsDuffy Polynomial	56.81	65.67
AAP Polynomial	56.94	65.76
AAPD Polynomial	57.02	**65.95**
AD Polynomial	**57.25**	65.92
TSAAPD Polynomial	56.43	65.32

The results of applying polynomial composite kernels reveal that these methods work better than their linear counterparts. On Timebank, AD polynomial composite kernel achieved the best result (i.e., over 6.2% improvement). On the other hand, on OTC, AAPD gained the best results with 3.45% improvement.

Unfortunately there are not a lot of researches on pattern based event-event relation classification, and we have to compare our work only with Mani algorithm. Regarding the hardness of the problem, it can be said, that the improvement is considerable.

6 Conclusion

In this paper, we have addressed the problem of extracting temporal relations between events, which has been a topic of interest since early days of natural language processing. Although syntactic features seem to be potentially useful in various text classification tasks, they have not yet been effectively exploited in temporal relation classification. We have tried to take advantage of such features to enhance classification performance. Support Vector Machines (SVM) has been chosen as our classification algorithm, due to its promising results in related works. Using SVM, two types of composite kernels have been proposed by combining convolution tree kernels and a simple event kernel.

The results of applying the new method, without using any extra annotated data, show a noticeable improvement over related works in the area of pattern based methods (including the state of the art method) in terms of accuracy.

It seems that using dependency structure of sentences or creating better kernels for SVM might be even further improve the accuracy of system.

References

Abe, S., K. Inui, and Y. Matsumoto. 2008. Two-phased event relation acquisition coupling the relation-oriented and argument-oriented approaches. In *Proceedings of the 22nd International Conference on Computational Linguistics*, pp.1-8.

Allen, J. F. 1984. Towards a general theory of action and time. *Artificial Intelligence*, 23(2):123-154.

Alonso, R. 2009. The value of time in unstructured data for IR. *In CIDR*.

Boser, B. E., I. M. Guyon, and V. N. Vapnik. 1992. A training algorithm for optimal margin classifiers. In *Proceedings of 5th workshop on Computational learning theory*, pp.144-152.

Chambers, N., S. Wang, and D. Jurafsky. 2007. Classifying temporal relations between events. In *Proceedings of ACL-45*, pp.173-176.

Chang, C. C. and C. J. Lin. 2001. Libsvm: a library for support vector machines. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

Chklovski, T. and P. Pantel. 2005. Global path-based refinement of noisy graphs applied to verb semantics. In *Proceeding of Joint Conference on Natural Language Processing*, pp.792-803.

Collins, M. and N. Duffy. 2001. Convolution kernels for natural language. In *Advances in Neural Information Processing Systems 14*, pp.625-632. MIT Press.

Cortes, C. and V. Vapnik. 1995. Support-vector networks. *Machine Learning*, pp.273-297.

Harris, Z. 1968. *Mathematical structure of language*. John Wiley Sons, New York.

Inui, T., K. Inui, and Y. Matsumoto. 2003. What kinds and amounts of causal knowledge can be acquired from text by using connective markers as clues? In *Proceedings of 6th International Conference on Discovery Science*, pp.180-193.

Khayyamian, M., S. A. Mirroshandel, and H. Abolhassani. 2009. Syntactic tree-based relation extraction using a generalization of Collins and Duffy convolution tree kernel. In *Proceedings of Human Language Technologies*, pp.66-71.

Lapata, M. and A. Lascarides. 2006. Learning sentence-internal temporal relations. *Journal of Artificial Intelligence Research*, 27:85-117.

Lin, D. and P. Pantel. 2001. Dirt-discovery of inference rules from text. In *Proceedings of the 7th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp.323-328.

Mani, I., B. Wellner, M. Verhagen, and J. Pustejovsky. 2007. *Three approaches to learning tlinks in timeml.* Technical Report CS-07-268. Computer Science Department, Brandeis University, Waltham, USA.

Mani, I., V. Marc, B. Wellner, C. M. Lee, and J. Pustejovsky. 2006. Machine learning of temporal relations. In *Proceedings of ACL-44*, pp.753-760.

Mirroshandel, S. A., M. Khayyamian, and G. R. Ghassem-Sani. 2009. Event-Time Temporal Relation Classification Using Syntactic Tree Kernels. In *Proceedings of the LTC'09*, Poznan, Poland (To appear).

Pekar, V. 2006. Acquisition of verb entailment from text. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pp.49-56.

Pustejovsky, J., P. Hanks, R. Sauri, A. See, R. Gaizauskas, A. Setzer, D. Radev, B. Sundheim, D. Day, L. Ferro, and M. Lazo. 2003. The TIMEBANK corpus. In *Proceedings of Corpus Linguistics 2003*, pp.647-656.

Szpektor, I., H. Tanev, and I. Dagan. 2004. Scaling web-based acquisition of entailment relations. In *Proceedings of EMNLP*, pp.41-48.

Tatu, M. and M. Srikanth. 2008. Experiments with reasoning for temporal relations between events. In *Proceedings of the 22nd International Conference on Computational Linguistics*, pp.857-864.

Torisawa, K. 2006. Acquiring inference rules with temporal constraints by using Japanese coordinated sentences and noun-verb co-occurrences. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pp.57-64.

Zhang, M., J. Zhang, J. Su, and G. D. Zhou. 2006. A composite kernel to extract relations between entities with both flat and structured features. In *Proceedings of the 21st International Conference on Computational Linguistics and ACL-44*, pp.825-832.